

# Weekly Report

11/24/2014-11/30/2014

## Reserach

本周，增加了一些 Matlab 代码结果的精确性。

1. 当时间片段变多，输出的值会变小  
比如输入的时如下 X:

$$X = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

原来应该有 100% 概率转移的地方，值的大小只有 0.85

2. 当人数变多，并且空帧（没有数据）较多时，输出的值也会变小

$$X1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} X10 = X9 = \dots X3 = X2 \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2)$$

输出的转移矩阵 A 的值，大概只有 0.5 左右

这时由于在迭代过程中，对 A 赋值时： $A_{ij}^{t*} = (\text{sign}(S_j - \lambda) * \lambda - S_j) / b_j$ , if  $|S_j| > \lambda$  在实际计算过程中，发生了  $\lambda$  和  $S_j$  的值非常相近的情况，导致计算结果不精确。这个问题，[2] 的作者在文章中也有提到，因为他只需要得到有关系的几张图，值的大小对他没有太大意义，然而我们对人数有比较高的要求。所以我又找到算法中使用的 Shooting Algotirhm[1].

$$\hat{\beta}_j = \begin{cases} \frac{\lambda - S_0}{2\mathbf{x}_j^T \mathbf{x}_j} & \text{if } S_0 > \lambda \\ \frac{-\lambda - S_0}{2\mathbf{x}_j^T \mathbf{x}_j} & \text{if } S_0 < -\lambda \\ 0 & \text{if } |S_0| \leq \lambda, \end{cases}$$

发现这和我们使用的迭代算法非常像。  
我们的算法中的  $S_j, b_j$  是这样得到：

$$S_j = \frac{2}{T} \sum_{t=2}^T w^t(i) \left( \sum_{k \neq j} A_{ik}^{t*} x_k^{t-1} - x_i^t \right) x_j^{t-1}$$

$$b_j = \frac{2}{T} \sum_{t=2}^T w^t(i) x_j^{t-1} x_j^{t-1}$$

因为这只是对一个人进行计算的，为了适应多人，根据[2]中的优化公式

$$A_{d*}^t = \operatorname{argmin} \sum_{l=1}^L \sum_{i=2}^T w^t(i) (x_{i,d}^l - A_{d*}^t x_{i-1}^l)^2 + \lambda \|A_{d*}^t\|$$

，对上面两个进行改写。

$$S_j = \frac{2}{TL} \sum_L \sum_{t=2}^T w^t(i) \left( \sum_{k \neq j} A_{ik}^{t*} x_k^{t-1} - x_i^t \right) x_j^{t-1}$$

$$b_j = \frac{2}{TL} \sum_L \sum_{t=2}^T w^t(i) x_j^{t-1} x_j^{t-1}$$

于是，问题也在这里产生：

对于问题 1，当时间片段变多，输出的值会变小。这是因为时间片段变多，T 变大，然而  $\sum_i w^t(i) = 1$ ，再除以 T，会使  $S_j$  变小。

对于问题 2，当人数变多，并且空帧（没有数据）较多时，输出的值也会变小。这是因为，我曾经为了消除空帧的影响，把空帧时的数据不加上去。

$$A_{i*}^{t*} = \operatorname{argmin} \frac{1}{TL} \sum_L \sum_{t=2}^T f(t) w^t(i) (x_i^t - A_{i*}^{t*} x^{t-1})^2 + \lambda \|A_{i*}^{t*}\|$$

$$\text{where, } f(x) = \begin{cases} 1 & \text{otherwise} \\ 0 & X(:, t) \text{ is a zero vector} \end{cases} \quad (3)$$

数据是没有加上去，但是还是除以了总人数 L，整体被拉低了。所以，不能除 T，不能除以太大的 L。最终使用如下公式（示意，实际公式还要考虑上面的 f(t) 等参数），

$$S_j = \frac{2}{L(i)} \sum_L \sum_{t=2}^T w^t(i) \left( \sum_{k \neq j} A_{ik}^{t*} x_k^{t-1} - x_i^t \right) x_j^{t-1}$$

$$b_j = \frac{2}{L(i)} \sum_L \sum_{t=2}^T w^t(i) x_j^{t-1} x_j^{t-1}$$

L(i)表示，在 i 时刻有数据的人数。

最终的结果是，解决了问题 1，部分解决问题 2（由 0.5 提升到 0.9）。问题 2 的残余是，i 时刻空帧，会对 i+1 时刻产生影响，L(i+1)的值还得根据前一时刻的空帧，再减减减。

## Plan for next week

- 继续 GPU 加速，下周再实现一些 matlab 的函数，就可以把代码往 cuda 上搬
- 阅读统计学习第 9 章

## References

- [1] Wenjiang J Fu. Penalized regressions: the bridge versus the lasso. *Journal of computational and graphical statistics*, 7(3):397--416, 1998.
- [2] Gunhee Kim and Eric P Xing. Reconstructing storyline graphs for image recommendation from web community photos.